中山大學
SUN YAT-SEN UNIVERSITY

国家超级计算广州中心
NATIONAL SUPERCOMPUTER CENTER IN GUANGZHOU

Computer Architecture

计 算 机 体 系 结 构

第13讲：Memory（3）

张献伟

xianweiz.github.io

DCS3013, 11/16/2022

Review Questions

- What is ‘tag’ in cache access?

  Part of address to be used to decide the access is hit/miss.

- Cache associativity?

  Cache is organized as sets, each of which contains multi blocks.

- Disadvantages of higher associativity?

  Larger tags, and higher overhead on tag comparator and data mux.

- Write back and write through?

  Write back first writes into cache, and then got updated into mem when being evicted; write through directly updates mem.

- Suppose memory capacity is 4GB, cache is 16KB with 16B block and 8 ways. Split address into tag-index-offset.

  Addr: 32b (byte addressable of 4GB)

  Offset: 4b (block is 16B), sets: 128 = 16KB/16B/8 -> Index: 7b

  Tag: 21b = 32 – 7 - 4

Evaluation Metrics[评价指标]

- Cache hit **ratio**
  - (# hits) / (# hits + # misses) = (# hits) / (# accesses)
- Average memory access time (AMAT)
  - (hit-ratio * hit-latency) + (miss-ratio * miss-latency)
- Cache hit **rate**: # of misses per kilo instructions (MPKI)

Example: Assume that
Processor speed = 1 GHz (1 n.sec. clock cycle)
Cache access time = 1 clock cycle
Miss penalty = 100 n.sec (100 clock cycles)
I-cache miss ratio = 1%, and D-cache miss ratio = 3%
74% of memory references are for instructions and 26% for data

Effective cache miss ratio = 0.01 * 0.74 + 0.03 * 0.26 = 0.0152
Av. (effective) memory access time = 1 + 0.0152 * 100 = 2.52 cycles = 2.52 n.sec

Cache Misses: 3 Cs

- **Compulsory**/Cold[强制性未命中]
  - First access to a block which is not in the cache
    - The block must be brought into the cache
  - Cache size does not matter
  - Solution: prefetching
- **Capacity**[容量性未命中]
  - Cache cannot contain all bocks needed during program execution
    - Blocks are evicted and later retrieved
  - Solution: increase cache size, stream buffers
- **Conflict**[冲突性未命中]
  - Occurs in set associative or direct mapped caches when too many blocks are mapped to the same set
  - Solution: increase associativity, victim cache
    - No conflict misses in fully associative cache

Coherence (invalidation) misses: other process updates memory

Cache Misses: 3 Cs (cont.)

- Mark D. Hill (https://pages.cs.wisc.edu/~markhill/)
  - computer architecture, parallel computing
  - memory systems, and performance evaluation
- The inventor of the widely-used 3C model of cache behavior (compulsory, capacity, and conflict misses)
- Simulation infrastructures
  - Multifacet's General Execution-driven Multiprocessor Simulator (GEMS) Toolset, 2005.
  - The gem5 simulator 2011.

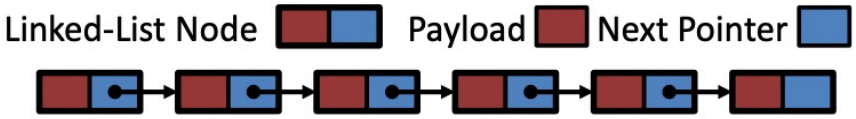
- ASPLOS系列专访—Mark D. Hill 谈体系结构：中国正从跟随者到创新领导者, https://zhuanlan.zhihu.com/p/26502720

Extra: Pointer Chasing

- A systematic micro-benchmarking approach for obtaining the hardware characteristics
  - Array elements are initialized with the index of the next memory access
  - The complete array is traversed sequentially

Table 1: Parameters of cache and P-chase test

Sym.	Description	Sym.	Description
C	cache capacity (unit: *int*)	N	array size (unit: *int*)
b	cache line size (unit: *int*)	s	stride size (unit: *int*)
a	cache associativity	M	#accesses to the array
T	number of cache sets	r	cache miss ratio



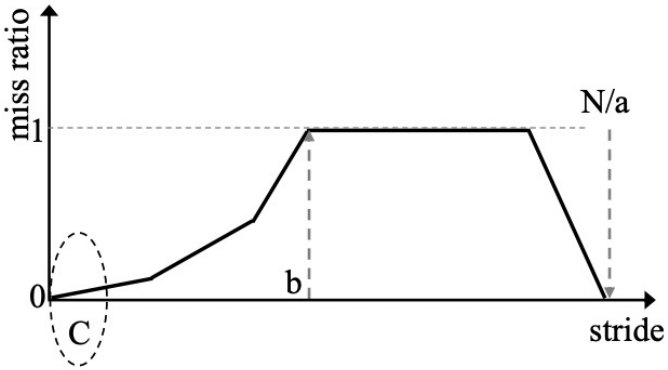
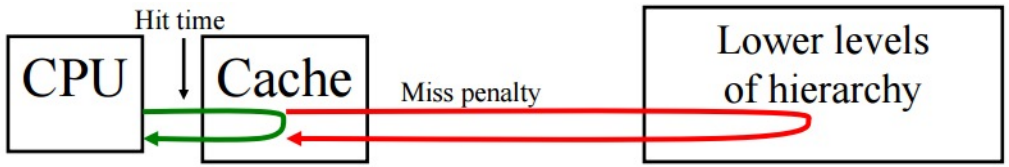


Figure 2: With varying traverse strides onto different sized arrays, P-chase microbenchmark exposes changing and predictable miss ratios, which reflect the cache configuration details.

Improve Cache Performance[性能提升]

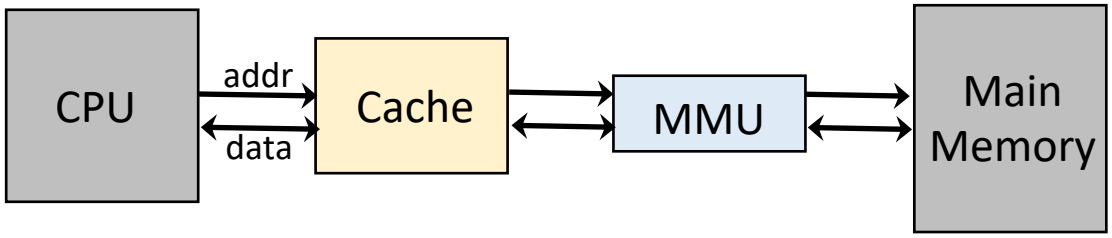
- Reduce the **miss ratio**
  - Larger block size
  - Larger caches
  - Higher associativity
    - But increase hit time and power consumption



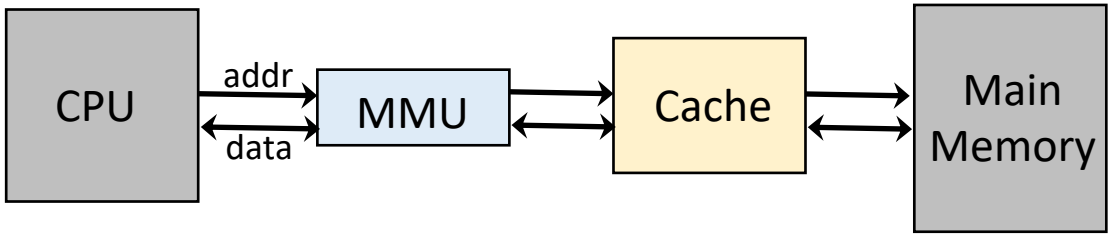
- Reduce the **miss penalty**
  - Multi-level caches
  - Read priority over write on miss
    - Serve reads before writes have completed
- Reduce **hit time**
  - Avoiding address translation
    - Just use virtual address

Virtual vs. Physical Caches

- Cache works on virtual addresses



- Cache works on physical addresses

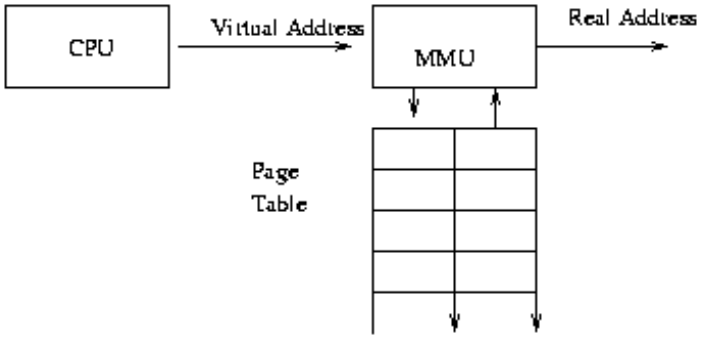


Virtual Memory[虚拟内存]

- Idea: give the programmer the illusion of a large address space while having a small physical memory
  - So that the programmer does not worry about managing physical memory
  - Virtual memory enables each process to have its own unique view of a computer's memory
- **Physical memory** is a storage hardware, made up of physical memory devices, which is organized as an array of *M* contiguous byte-sized cells
  - Each byte has a unique physical address
- Physical vs. virtual address
  - Physical addresses are unique in the system, only used by kernel
  - Virtual memory addresses are unique per-process, used by userspace programs

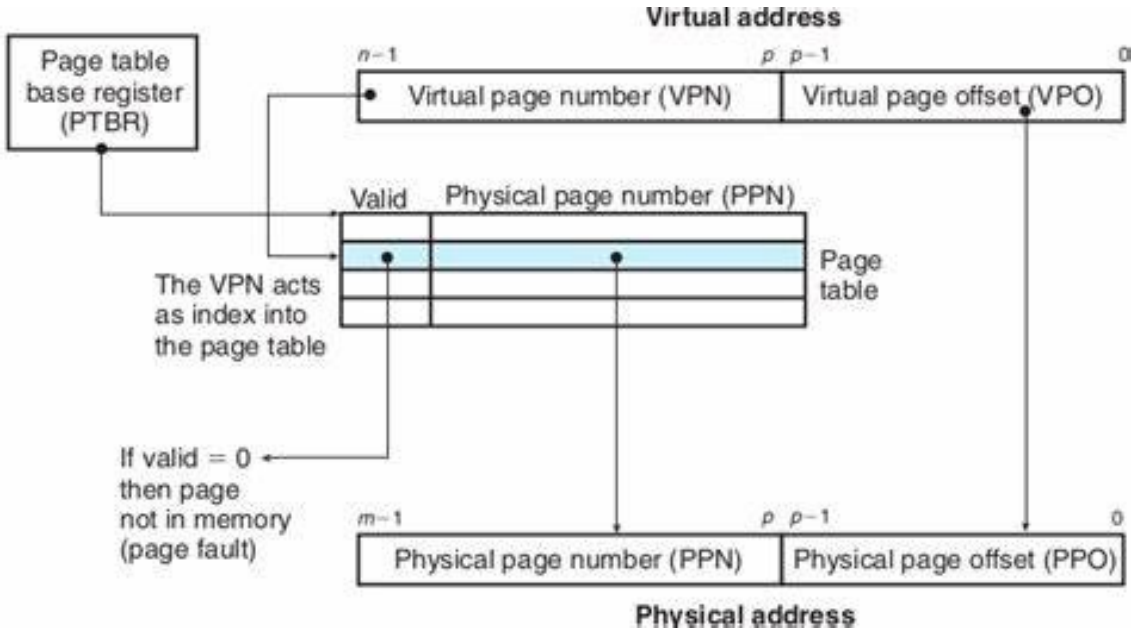
Address Translation[地址转换]

- Address Translation: the hardware converts virtual addresses into physical addresses via an OS-managed lookup table (page table)
- HW and SW cooperatively manage the translation
  - OS software
  - Address translation hardware in MMU
  - Page tables stored in physical memory or disk
- Memory management unit[内存管理单元]
  - Includes Page Table Base Register(s), TLBs, page walkers



Address Translation (cont.)

- A virtual page is mapped to
  - A physical frame, if the page is in physical memory
  - A location in disk, otherwise
- If an accessed virtual page is not in memory, but on disk
  - Virtual memory system brings the page into a physical frame and adjusts the mapping → this is called *demand paging*

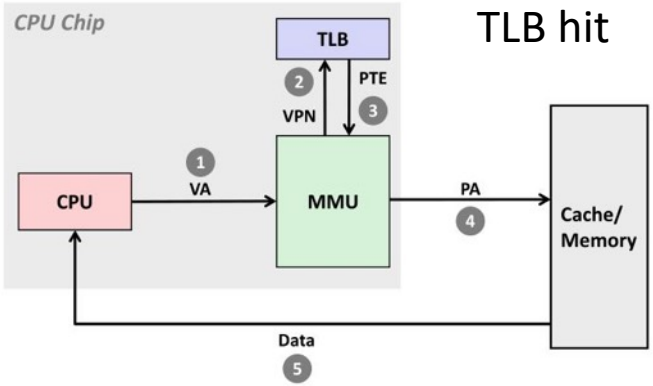


Page Table and TLB[页表]

- Page table is the table that stores the mapping of virtual pages to physical frames
- Page table is just a data structure to map VA (or really VPN) to PA (PFN)
  - Each process has its own set of page tables
  - Page table size for a process is roughly 4MB for 32-bit address space with 4-byte page table entry (PTE)
    - Can be 400MB for 100 processes
- TLB: part of chip's MMU to speed address translation
  - Cache the popular virtual-to-physical address translations
  - Upon each virtual memory reference, the hw first checks the TLB to see if the desired translation is held there
    - If so, the translation is performed without having to consult the page table (which has all translations)

TLB (cont.)

- A typical TLB might have 32, 64, 128 entries, which are *fully associative*
- TLB contains v2p translations that are only valid for the currently running process
  - Those translations are not meaningful for other processes
  - Flush is needed when switching from one process to another
- Accesses to virtual addresses not listed in TLB (a “TLB miss”) trigger a page table lookup
  - Performed either by hw or the page fault handler

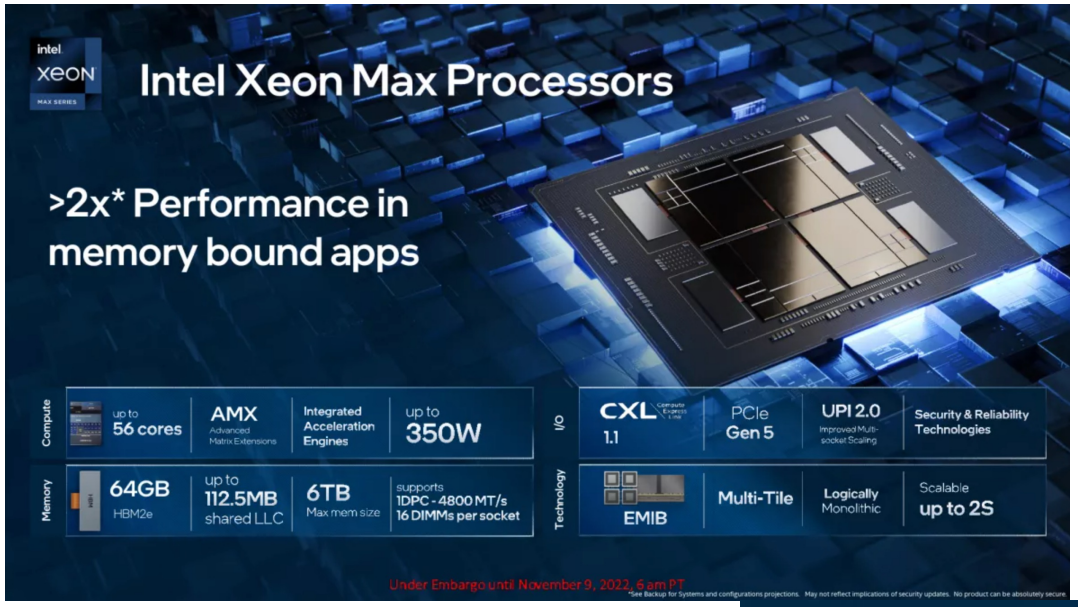


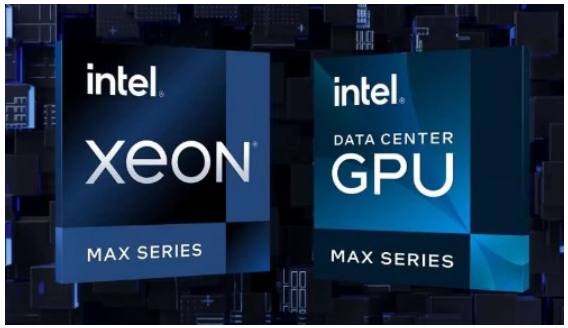
Page Fault[页缺失]

- Physical memory is a cache for pages stored on disk
  - In fact, it is a *fully associative* cache in modern systems (a virtual page can be mapped to any physical frame)
- Page fault: a DRAM cache miss
  - Find out where the contents of the page are stored on disk
  - Possible that this page isn't anywhere at all
    - The memory reference is buggy and thus the process will be killed
- Suppose page fault happens on page *p1*, which is on disk
  - Find page *p2* mapped to some frame *f* that is not used much
  - Copy the contents of frame *f* out to disk
  - Clear page *p2*'s valid bit (subsequent refs to *p2* will cause page faults)
  - Update the MMU's table so that *p1* is mapped to frame *f*
  - Return from the interrupt, allowing the CPU to retry the inst that caused the interrupt

Fore more on VM, see https://pages.cs.wisc.edu/~remzi/OSTEP/

Intel Xeon Max

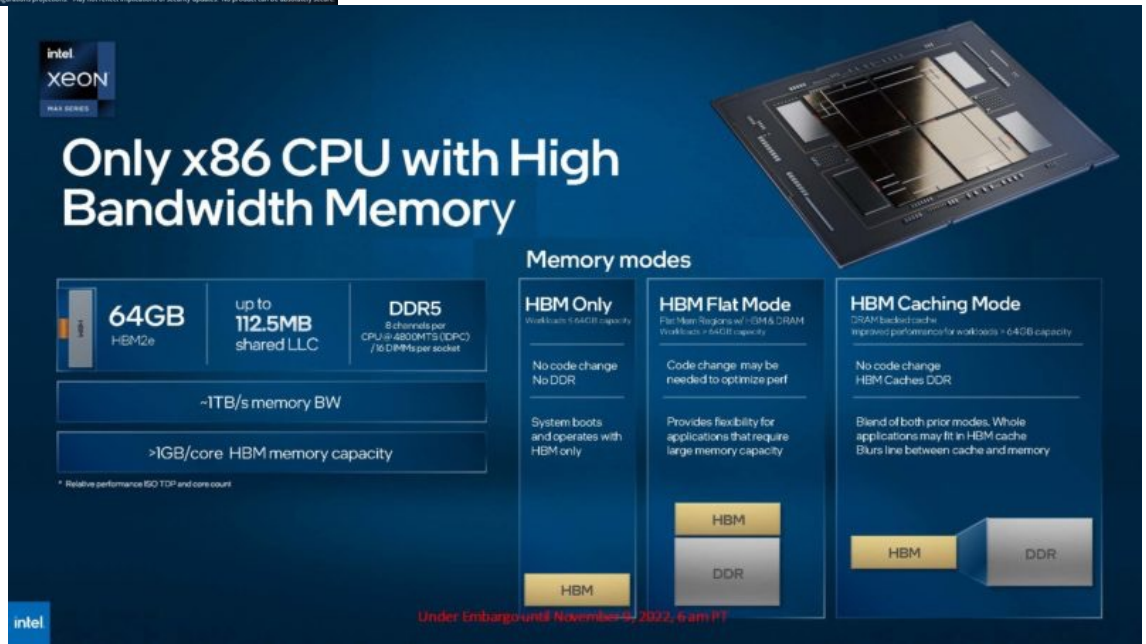




https://www.servethehome.com/intel-xeon-max-cpu-is-the-sapphire-rapids-hbm-line/

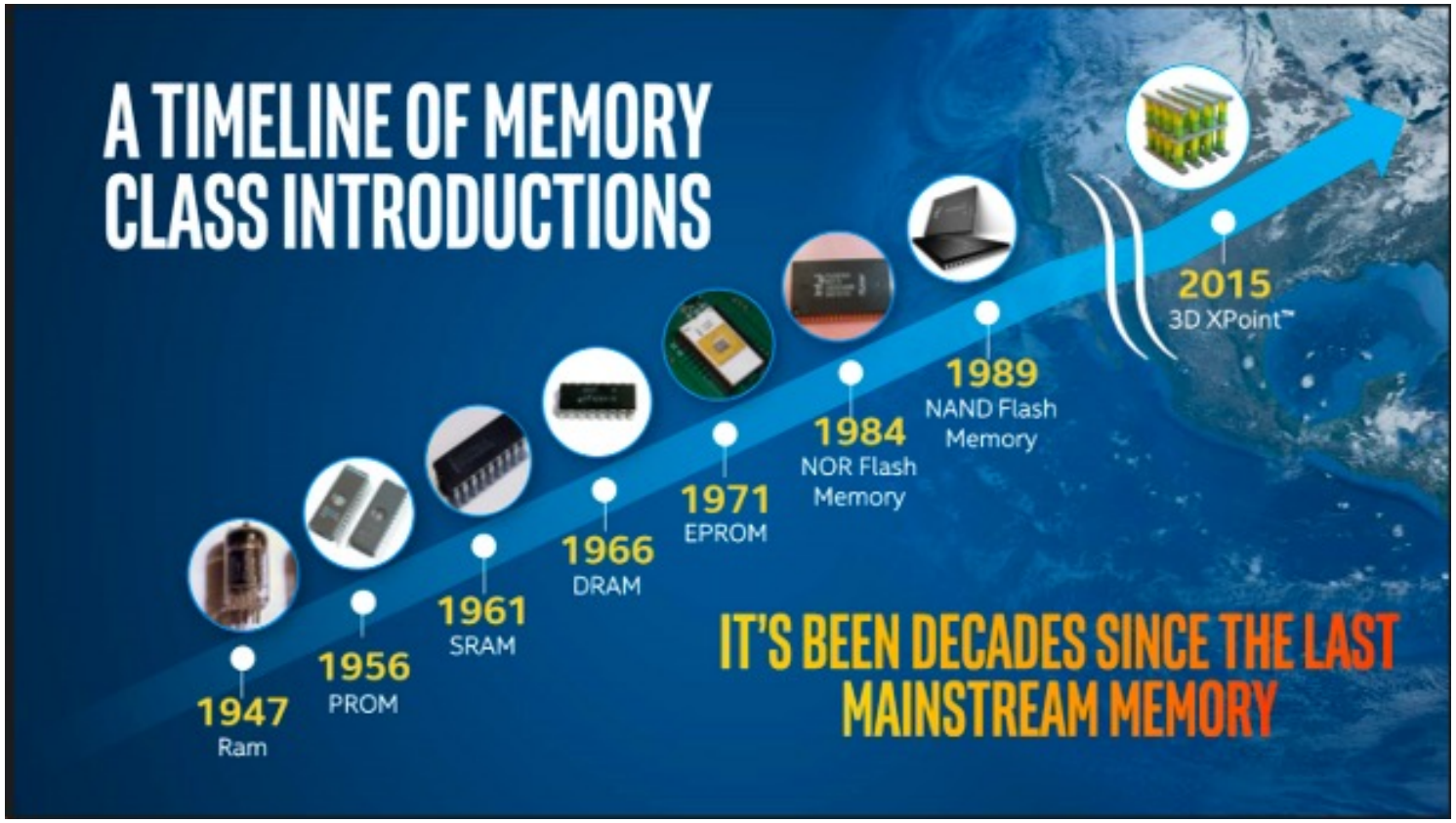
https://www.tomshardware.com/news/intel-fires-up-xeon-max-cpus-gpus-to-rival-amd-nvidia

https://www.intel.com/content/www/us/en/newsroom/news/introducing-intel-max-series-product-family.html#gs.iub4p3



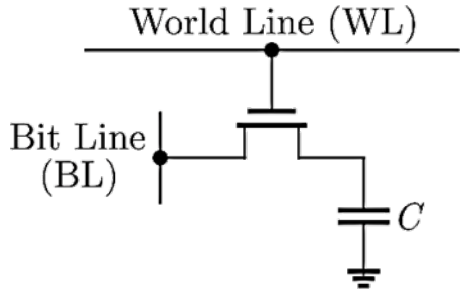
Memory Technology[存储技术]

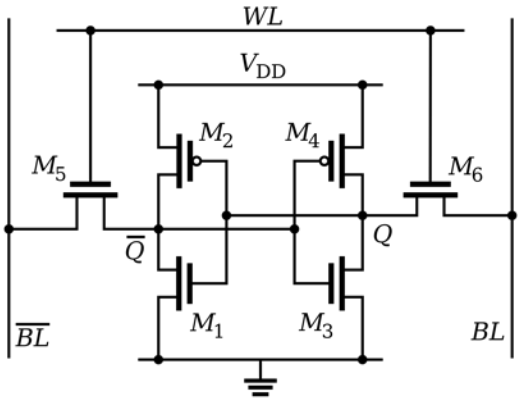
- Performance of main memory
  - Latency: affects Cache Miss Penalty
  - Bandwidth: affects I/O & Large Block Miss Penalty

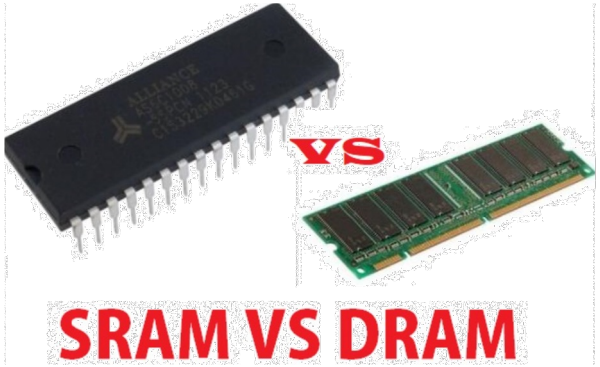


DRAM vs. SRAM

- Main Memory uses *DRAM*: Dynamic Random Access Memory
  - Needs to be refreshed periodically (one row at a time)
  - Addresses divided into 2 halves (memory as a 2D matrix):
    - *RAS* or *Row Access Strobe*
    - *CAS* or *Column Access Strobe*
- Cache uses *SRAM*: Static RAM
  - No refresh (6 transistors/bit vs. 1)
    - *Size*: DRAM/SRAM *4-8*
    - *Cost/Cycle time*: SRAM/DRAM *8-16*



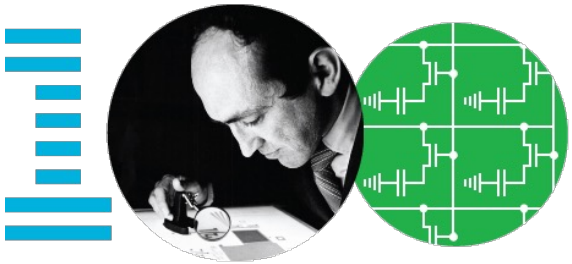




DRAM

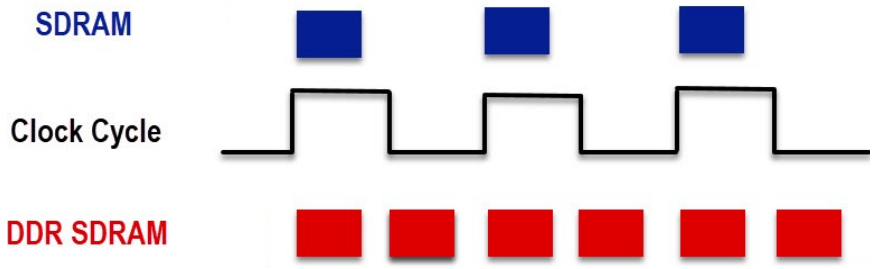
- History
  - 1966: Invented by Robert Dennard of IBM
  - 1967: DRAM patent was filed (issued 1968)
  - 1970: Intel built 1Kb DRAM chip (3T cell)
  - ~1975: 4Kb DRAM chip (1T cell)

Dennard Scaling Law: as transistors shrank, so did necessary voltage and current; power is proportional to the area of the transistor



"I knew it was going to be a big thing, but I didn't know it would grow to have the wide impact it has today."

- SDRAM = DRAM with a clocked interface
- DDR SDRAM = double data rate, transfer data at both clock edges
  - DDR1 (2.5 V, 200-400 MHz)
  - DDR2 (1.8 V, 400-1066 MHz)
  - DDR3 (1.5 V, 800-2133MHz)
  - DDR4 (1.2 V, 1600-5333 MHz)
  - DDR5 (1.1V, 3200-6400 MHz)



https://www.ibm.com/ibm/history/ibm100/us/en/icons/dram/